44. The computer program product of claim 43, wherein the database meta-information generator class encapsulates information identifying the structure of the relational database and the one or more delete actions into the class object.

45. The computer program product of claim 44, wherein the one or more delete actions is at least one of cascade delete and nullify columns delete.

46. A method of generating a class for deletion of data representations of objects in a relational database, comprising:

determining a structure of the relational database;

determining one or more default delete actions based on the structure of the relational database;

receiving user input to modify the one or more default delete actions; and

generating the class object based on the determined structure, the determined one or more delete actions and the user input.

47. The method of claim 46, wherein the user input overrides one or more of the one or more default delete actions.

48. The method of claim 46, wherein the user input inserts one or more delete action constraints.

REMARKS

Claims 1, 3-12 and 14-48 are pending in the present application. By this Response, claims 1 and 12 are amended to incorporate the subject ~~matter of canceled~~ claims 2 and 13. Claims 3, 7, 14 and 17 are amended to correct their dependency based on the cancellation of claims 2 and 13. Reconsideration of the claims is respectfully requested.

I. 35 U.S.C. § 102, Alleged Anticipation Based on Salo

The Office Action rejects claims 1-4, 7, 12-15, 17, 20-21, 24, 27-28, 35-36, 43-44 and 46 under 35 U.S.C. § 102(e) as being anticipated by Salo et al. (U.S. Patent No. 6,456,995). This rejection is respectfully traversed.

As to independent claim 1, the Office Action states:

> With respect to claim 1, Salo discloses determining a structure of the relational database (col. 4, lines 49-57 and fig. 1, item 100); determining a delete action based on the structure of the relational database (col. 7, lines 60-67 and col. 8, lines 1-3); generating database modification commands based on the determined delete action (col. 6, lines 60-67, col. 7, lines 1-4, also col. 8, lines 41-51, see fig. 4 and fig. 6); and sending the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands (co. 6, lines 26-41 and col. 7, lines 9-17).

Office Action dated November 7, 2002, pages 2-3.

Claim 1, which is representative of claim 12 with regard to similarly recited subject matter, reads as follows:

> 1. A method of deleting object data from a relational database, comprising:
>
> determining a structure of the relational database, wherein the determining the structure of the relational database includes invoking a database meta-information class object associated with the relational database;
>
> determining a delete action based on the structure of the relational database;
>
> generating database modification commands based on the determined delete action; and
>
> sending the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands. (emphasis added)

A prior art reference anticipates the claimed invention under 35 U.S.C. § 102 only if every element of a claimed invention is identically shown in that single reference, arranged as they are in the claims. *In re Bond*, 910 F.2d 831, 832, 15 U.S.P.Q.2d 1566, 1567 (Fed. Cir. 1990). All limitations of the claimed invention must be considered when

determining patentability. *In re Lowry*, 32 F.3d 1579, 1582, 32 U.S.P.Q.2d 1031, 1034 (Fed. Cir. 1994). Anticipation focuses on whether a claim reads on the product or process a prior art reference discloses, not on what the reference broadly teaches. *Kalman v. Kimberly-Clark Corp.*, 713 F.2d 760, 218 U.S.P.Q. 781 (Fed. Cir. 1983). Salo does not identically show each and every element of the claimed invention as discussed hereafter.

Salo is directed to a system, method and computer program produce for ordering objects for insertion to and deletion from a relational database. With the system of Salo, objects that are subject to a database operation are first placed in an unsorted object list. The sorted object list is initially empty. For each object in the unsorted object list, its associations with other objects are identified and placed in an associations list (see Figure 9). When an object's associations are being identified, the current object is added to the end of the sorted objects list. For each association, a test is made as to whether the current object has either insert precedence (if the object is to be inserted into the database) or delete precedence (if the object is to be deleted from the database) over the current association. If the current object has precedence over the current association and the associated object or objects are already included in the sorted object list in a position before the current object, then the associated object and its lower precedence associated objects are recursively moved to the end of the sorted object list.

Salo does not teach determining a structure of a relational database which includes invoking a database meta-information class object associated with the relational database. The Office Action alleges that this feature is taught by Salo at column 4, lines 45-67 and column 7, lines 9-67. Column 4, lines 45-67 merely teaches that the relational database system 100 may include a relational database 110, one or more mainframe, midrange and/or personal computers and relational database 110 may be a single relational database or multiple relational databases that reside on one or more data processors. Example relational databases that may be used include the DB2 and SQL computer program products of IBM Corporation. There is no teaching of a meta-information class object associated with a relational database in this section of Salo.

Column 7, lines 9-67 merely teach the ordering of objects that are the subject of database operations, e.g., insertion or deletion. There is no teaching in column 7, lines 9-67 regarding the determination of a structure of a database using a meta-information class

object associated with the database, as recited in claim 1. To the contrary, as described in column 9, lines 10-42, each object in an unsorted object list is examined to identify its own associations to determine a precedence for insertion and deletion. There is no consulting of a meta-information class object to determine the structure of a relational database in order to determine delete actions to be performed. In view of the above, Applicants respectfully submit that Salo does not teach each and every feature of independent claim 1 and similar features in independent claim 12.

With regard to independent claims 20, 27 and 35, the Office Action states that Salo teaches the features of these claims at column 4, lines 49-57; figure 1, item 100; column 6, lines 26-41; column 6, line 60 to column 7, line 4; column 7, lines 9-17; column 7, line 60 to column 8, line 3; column 8, lines 41-51; and figures 4 and 6. None of these sections of Salo teach generating a class object based on a determined structure and one or more determined delete actions, as recited in claim 20 and similar features of claims 27 and 35. Each of these sections will be address in the above order.

Column 4, lines 49-57 read as follows:

> As shown in FIG. 1, a relational database system 100 includes a relational database 110. The relational database system 100 may include one or more mainframe, midrange and/or personal computers and the relational database 110 may be a single relational database or multiple relational database that reside on one or more data processors. Example relational databases include the DB2 and SQL computer program products of IBM Corp., the assignee of the present invention.

All this section of Salo teaches is that a relational database system may have a plurality of computers of varying types and one or more relational database that reside on one or more data processors. There is nothing in this section of Salo that teaches generating a class object based on the determined structure and the determined one or more delete actions, as recited in claim 20 and similar features in claims 27 and 35.

Figure 1, item 100 is a relational database system having a relational database 110. There is nothing in Figure 1 that teaches determining a structure of the relational database, determining one or more delete actions based on the structure of the relational

database, or generating a class object based on the determined structure and the determined one or more delete actions.

Column 6, lines 26-41 reads as follows:

> The database referential integrity constraints do not generally map to the logical parent-child object relationships in a consistent manner, because referential integrity rules are generally enforced based on the key information and not based on logical relationships. For example, association pointing from employee to address may be implemented in the database as a foreign key reference from employee to address or from address to employee. Furthermore, the relationships between objects generally are independent of the order in which the objects are stored to or deleted from the database. Therefore, the order in which trees of objects need to be stored to or deleted from the relational database generally cannot be deduced either from the event flow in the transaction processing system 130 or from the object relationship logical directions.

There is nothing in this section of Sato that teaches generating a class object based on the determined structure and the determined one or more delete actions. To the contrary, this section, and the subsequent paragraph at column 6, lines 42-48, of Sato merely teach that it is not possible to know the structure of the database, e.g., the order in which trees of objects need to be stored to or deleted from the relational database, based on the event flow of object relationship logical directions. This section of Salo has nothing to do with generating class objects, let alone generating class objects based on a determined structure of a relational database and one or more determined delete actions.

Column 6, line 60 to column 7, line 4 reads as follows:

> Since the relationship between departments and employees is one to many, the department Table A includes a primary key and the employee Table B includes a foreign key that refers to the appropriate primary key of department Table A. Stated differently, the relationship between Table A and Table B is owner to member so that the foreign key in the member table refers to the appropriate primary key in the owner table. In contrast, the relationship between employee Table B and address Table C is one to one so that the relationship may be defined using a foreign key in the employee Table B that refers to the address Table C as shown by the solid arrow, or a foreign key in the address Table C that refers to the employee Table B as shown by the dotted arrow.

All this section of Salo teaches is the use of primary and foreign keys in tables of a relational database. While the tables of an exemplary relational database are described in order to illustrate relationships in conventional relational databases, there is no teaching in this section of Salo to generate a class object based on the determined structure and the determined one or more delete actions.

Column 7, lines 9-17 read as follows:

> Database operation ordering 150 according to the present invention provides a mechanism for automatically ordering the relational insert and delete operations according to the referential integrity constraints defined for the relational database 110 when trees of objects are to be stored to or deleted from the database. The ordering mechanism preferably utilizes the information of how the relationships between the objects are mapped to the primary-key/foreign-key pairs in the database and integrity rules for the pairs.

While this section of Salo teaches that the ordering mechanism may utilize information of how the relationships between objects are mapped to the primary-key /foreign-key pairs, there is no teaching of determining one or more delete actions based on the structure of the relational database, or generating a class object based on the determined structure and the determined one or more delete actions.

Column 7, line 60 to column 8, line 3 reads as follows:

> Still referring to FIG. 4, the objects to be inserted into the relational database are ordered according to an insert precedence of the objects into the relational database to thereby define an insert order in an insert-list at Block 420. At Block 430, the objects to be deleted from the relational database are ordered according to a delete precedence of the objects from the relational database to thereby define a delete order in a delete-list. It will be understood that the operations of Blocks 420 and 430 may be performed in a reverse sequence from that shown. Details of performing these operations will be described below.

While this section of Salo teaches generating an ordering of a delete operation based on a delete precedence, there is nothing in this section that teaches to generate a class object based on a determined structure of a relational database and one or more

determined delete actions. There is no mention whatsoever regarding the generating of a class object in this section of Salo.

Column 8, lines 41-51 read as follows:

> The same operations can be performed for resolving the detailed prerequisites of insert and update operations. Accordingly, these operations are described generically in FIG. 6. However, it will be understood that these operations are performed separately for ordering the objects to be inserted into the relational database according to an insert precedence of the objects into the relational database to thereby define an insert order, and separately for ordering the objects to be deleted from the relational database according to a delete precedence of the object form the relational database to thereby define a delete order.

Again, there is no teaching in this section regarding the generating of an class object based on a determined structure of a relational database and one or more determined delete actions. All that this section teaches is that ordering of objects to be inserted and objects to be deleted are performed separately. There is no mention of generating a class object, let alone generating a class object based on a determined structure of a relational database and one or more determined delete actions.

Finally, Figures 4 and 6 of Salo provide flowcharts of the operation of the system of Salo. Figure 4 describes the clustering of objects according to their database operations, the ordering of objects in an insert list and a delete list, and the performance of the insertion, updating and deletion operations based on these ordered lists. There is nothing in Figure 4 that teaches generating a class object based on a determined structure of a relational database and one or more determined delete actions.

Figure 6 describes the methodology for traversing the unsorted object list and determining for each object in the list, the associations for that object. The flowchart further describes the entry of the object into the sorted list and the entry of associated objects into the sorted list based on the associations and an insertion/deletion precedence. There is nothing in Figure 6 of Salo that teaches generating a class object based on a determined structure of a relational database and one or more determined delete actions.

Thus, in view of the above, Applicants respectfully submit that Salo does not teach each and every feature of independent claims 20, 27 and 35 as is required under 35

U.S.C. § 102(e). Neither the sections cited by the Office Action, nor any other section of Salo, teach the feature of generating a class object based on a determined structure of a relational database and one or more determined delete actions.

Regarding independent claim 43, the Office Action points to the same sections of Salo discussed above with regard to the rejection of claims 20, 27 and 35. Claim 43 reads as follows:

> 43. A computer program product in a computer readable medium for generating a class object for deletion of data representations of objects in a relational database, comprising:
> a meta-information class for determining a structure of the relational database and one or more delete actions based on the structure of the relational database; and
> a database meta-information generator class for generating the class object based on the determined structure and the determined one or more delete actions. (emphasis added)

As detailed in the arguments above regarding claims 1 and 12, Salo does not teach a meta-information class or the use of such a class to determine a structure of a relational database. Furthermore, Salo does not teach that such a meta-information class can be used to determine one or more delete actions based on the structure of the relational database. In addition, as discussed above with regard to claims 20, 27 and 35, Salo does not teach generating a class object based on a determined structure of a relational database and one or more determined delete actions. Thus, Salo does not teach either of the features highlighted above in claim 43.

Regarding independent claim 46, the Office Action again points to the sections noted above with regard to claims 20, 27, 35 and 43. Claim 46 recites:

> 46. A method of generating a class for deletion of data representations of objects in a relational database, comprising:
> determining a structure of the relational database;
> determining one or more default delete actions based on the structure of the relational database;
> receiving user input to modify the one or more default delete actions; and

> generating the class object based on the determined structure, the determined one or more delete actions and the user input. (emphasis added)

None of the sections cited by the Office Action, as discussed previously, teach or suggest generating a class object based on a determined structure of a relational database and one or more delete actions. In addition, none of these cited sections, or any other section, of Salo teach receiving user input to modify one or more default delete actions or generating the class object based on the user input.

In view of the above, Applicants respectfully submit that Salo does not teach each and every feature of independent claims 1, 12, 20, 27, 35, 43 and 46 as is required under 35 U.S.C. § 102(e). At least by virtue of their dependency on these claims, Salo does not teach the features of dependent claims 3-4, 7, 13-15, 17, 21, 24, 28, 36 and 44, respectively. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 1-4, 7, 12-15, 17, 20-21, 24, 27-28, 35-36, 43-44 and 46 under 35 U.S.C. § 102(e).

In addition to the above, Salo does not teach the specific features recited in dependent claims 3-4, 7, 14-15, 17, 21, 24, 28, 36 and 44. For example, with regard to claims 3 and 14, Salo does not teach a database meta-information class object that encapsulates a dependency structure of the relational database. The Office Action alleges that this feature is taught in Salo at column 6, lines 42-64 which reads:

> Accordingly, at the end of an object-oriented transaction by transaction processing system 130, the objects generally are known, the operations that were performed are generally known and the data that results is generally known. However, the order or sequence in which changes should be made to the relational database 110 in order to preserve the referential integrity constraints are not known.
>
> There are generally three kinds of relational operations that can be performed on persistent objects: insert, update and delete. The dependency between these operation types are generally as follows:
>
> (1) Insert operations can be dependent on other inserts, but not on any other kind of operations. For example, data to be inserted can refer to other data to be inserted.
>
> (2) Update operations are not dependent upon other updates, but they can be dependent on inserts. For example, data is to be updated to refer to data to be inserted.

(3) Delete operations can be dependent upon other deletes. For example, data to be deleted refers to other data to be deleted. Delete operations also can be dependent on updates. For examples, data is to be updated to de-reference data to be deleted.

While this section of Salo teaches a dependency of operation types, there is nothing in this section of Salo that teaches database meta-information class object that encapsulates a dependency structure of the relational database. There is not even a mention of a meta-information class object, let alone the encapsulation of a dependency structure of a relational database in a meta-information class object.

Regarding claims 21, 28, 36 and 44, a similar distinction over Salo applies. That is, Salo does not teach a class object that encapsulates information identifying the structure of a relational database and one or more delete actions. The Office Action alleges that these features are taught at column 4, lines 45-67, column 6, lines 42-64 and column 7, lines 9-67 of Salo. Each of these sections of Salo have been addressed above and none of them teach a class object that encapsulates information identifying the structure of a relational database and one or more delete actions.

Regarding claims 4 and 15, Salo does not teach a database meta-information class object includes a delete action identifier for each dependent table of a plurality of table sin the relational database. The Office Action alleges that this feature is taught by Salo at column 5, lines 44-55. This section of Salo merely describes an example of the relationships between tables in a conventional relational database. There is nothing in this section, or any other section, of Salo that teaches a database meta-information class object includes a delete action identifier for each dependent table of a plurality of tables in the relational database.

With regard to claims 7 and 17, Salo does not teach that the meta-information class object is generated based on a file describing the structure and delete actions for tables in the relational database. The Office Action alleges that this feature is taught in Salo at column 5, lines 44-55 which have been discussed above. There is no teaching in this, or any other, section of Salo regarding a file describing the structure and delete actions for tables in the relational database or the generation of a meta-information class object from such a file. A similar distinction applies to claim 24 as well.

Thus, in view of the above, the rejected dependent claims 3-4, 7, 13-15, 17, 21, 24, 28, 36 and 44 are also allowable over Salo based on the specific features recited therein.

II. 35 U.S.C. § 103, Alleged Obviousness Based on Salo and Crus

The Office Action rejects claims 5-6, 9-11, 16, 18-19, 22-23, 25-26, 29-31, 33-34, 37-39, 41-42, 45 and 47-48 under 35 U.S.C. § 103 as being unpatentable over U.S. Patent No. 6,456,995 issued to Salo et al. (herein Salo) in view of U.S. Patent No. 4,947,320 issued to Crus et al. (hereinafter Crus). This rejection is respectfully traversed.

Applicants respectfully submit that, under 35 U.S.C. § 103(c), Salo cannot be relied upon as prior art under section 103. 35 U.S.C. § 103(c) states:

> (c) Subject matter developed by another person, which qualifies as prior art only under one or more of subsections (e), (f), and (g) of section 102 of this title, shall not preclude patentability under this section where the subject matter and the claimed invention were, at the time the invention was made, owned by the same person or subject to an obligation of assignment to the same person.

The instant application was filed after November 29, 1999 and thus, section 103(c) applies to the present application. The Salo patent qualifies as prior art only under 35 U.S.C. § 102(e). Furthermore, the instant application and the Salo patent were commonly owned or subject to an obligation of assignment to the same person at the time the invention in the present application was made. Therefore, the Salo patent cannot be used in a 35 U.S.C. § 103 rejection to preclude patentability. As such, the rejection of claims 5-6, 9-11, 16, 18-19, 22-23, 25-26, 29-31, 33-34, 37-39, 41-42, 45 and 47-48 under 35 U.S.C. § 103 is improper and should be withdrawn. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 5-6, 9-11, 16, 18-19, 22-23, 25-26, 29-31, 33-34, 37-39, 41-42, 45 and 47-48 under 35 U.S.C. § 103.

III. 35 U.S.C. § 103, Alleged Obviousness Based on Salo and Goodwin

The Office Action rejects claims 8, 32, and 40 under 35 U.S.C. § 103 as being unpatentable over U.S. Patent No. 6,456,995 issued to Salo et al. (herein Salo) in view of U.S. Patent No. 6,199,195 issued to Goodwin et al. (hereinafter Goodwin). This rejection is respectfully traversed.

As with the 35 U.S.C. § 103 rejection discussed above, Salo cannot be used in a rejection under 35 U.S.C. § 103 as stated in 35 U.S.C. § 103(c). Therefore, the rejection of claims 8, 32 and 40 under 35 U.S.C. § 103 based on Salo and Goodwin is improper and should be withdrawn. Accordingly, Applicants respectfully request withdrawal of the rejection of claims 8, 32 and 40 under 35 U.S.C. § 103.

IV. Conclusion

It is respectfully urged that the subject application is patentable over Salo, Goodwin and Crus and is now in condition for allowance. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

Respectfully submitted,

DATE: February 7, 2003

Stephen J. Walder, Jr.
Reg. No. 41,534
Carstens, Yee & Cahoon, LLP
P.O. Box 802334
Dallas, TX 75380
(972) 367-2001
Attorney for Applicant

APPENDIX OF CLAIM AMENDMENTS

Please cancel claims 2 and 13 without prejudice or disclaimer.

Please amend claims 1, 3, 7, 12, 14 and 17 as follows:

1. (Once Amended) A method of deleting object data from a relational database, comprising:

determining a structure of the relational database, wherein determining the structure of the relational database includes invoking a database meta-information class object associated with the relational database;

determining a delete action based on the structure of the relational database;

generating database modification commands based on the determined delete action; and

sending the database modification commands to a relational database server, wherein the relational database server deletes the object data from the relational database based on the database modification commands.

3. (Once Amended) The method of claim [2] 1, wherein the database meta-information class object encapsulates a dependency structure of the relational database.

7. (Once Amended) The method of claim [2] 1, wherein the database meta-information class object is generated based on a file describing the structure and delete actions for tables in the relational database.

12. (Once Amended) A system for deleting object data from a relational database, comprising:

a data processor; and

a relational database storage device, wherein the data processor determines a structure of the relational database, wherein the data processor determines the structure of the relational database by invoking a database meta-information class object associated with the relational database, determines a delete action based on the structure of the

relational database, generates database modification commands based on the determined delete action and sends the database modification commands to the relational database storage device, wherein the relational database storage device deletes the object data from the relational database based on the database modification commands.

14. (Once Amended) The apparatus of claim [13] 12, wherein the database meta-information class object encapsulates a dependency structure of the relational database.

17. (Once Amended) The apparatus of claim [13] 12, wherein the database meta-information class object is generated based on a file describing the structure and delete actions for tables in the relational database.